CHAPTER

3

KEY WINDOWS 2000 COMPONENTS

After reading this chapter and completing the exercises, you will be able to:

- Recount the basic start-up process for Windows 2000
- Understand the Windows 2000 Registry
- Appreciate the Windows 2000 base functionality
- Recognize basic administrative tools included with Windows 2000

Like most operating systems, Windows 2000 is made up of multiple files, each of which serves a specific purpose to create a workable computing environment. These files range from memory managers and hardware drivers to management components and interface translators. This chapter looks at the important files and components found in Windows 2000.

IMPORTANT FILES AND SUBSYSTEMS

Windows 2000 contains dozens of files that initiate its core components, which in turn load additional files that control such things as the user interface, security, and hardware support. These files help establish a workable runtime environment for Windows 2000. Each of these files is discussed in detail throughout this chapter.

Microsoft took a "divide and conquer" approach when designing Windows 2000 (and its predecessor, Windows NT). That approach shows clearly in the **modular architecture** depicted in Figure 3-1. The system is divided into two distinct operating modes: user mode and kernel mode. Each of these modes is further divided into single-function or special-purpose modules. Each module is independent, which means that program code is not shared across any two modules. In some instances, module independence increases the overall operating system size because of the need to maintain duplicate code. User and kernel modes were discussed in Chapter 2. On the other hand, this design offers improved reliability, performance, and fault tolerance, all of which outweigh the costs of a larger system footprint.

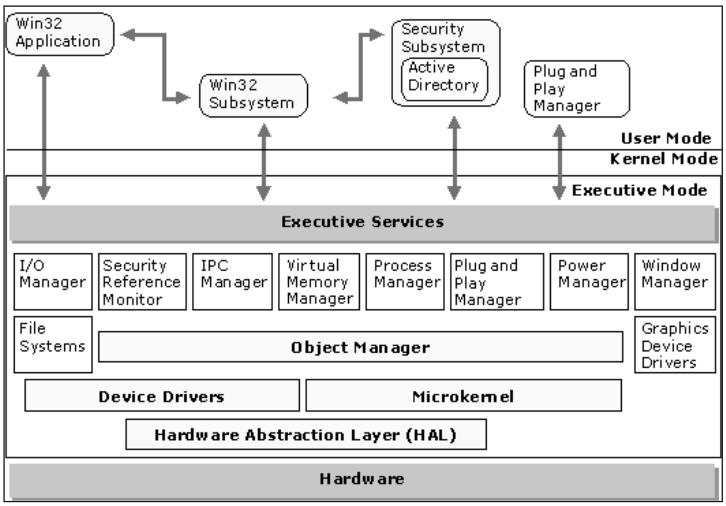

Figure 3-1 Windows 2000 architecture

A modular operating system design, like that characterizing Windows 2000, has several important benefits. Each module can be replaced or repaired without affecting other modules. As long as communication interfaces between modules remain the same (or functionally equivalent), each module can be replaced with either an updated version from Microsoft or a module from a third-party vendor. Modules cooperate to create networking, communications, and application environments, which in turn support tasks and applications involved in general computing activities.

Before you examine the details of the components in an active operating system, you need to explore the process by which the operating system is loaded. The following sections discuss

the **bootstrapping** (or just "boot") **files**, the **core system files**, and other key elements in a Windows 2000 system.

Bootstrapping

The **bootstrapping** process encompasses the initial activities that occur before an operating system can load. This process basically transforms a hunk of metal and plastic into a computing device ready to accept instructions.

The following initial activities make up the bootstrapping process:

- Powering on the computer: Bootstrapping first provides power to the computer, usually when a user flips the power button on the box.
- **Power On, Self-Test (POST)**: This internal diagnostic ensures that all required core hardware is present and operating properly. The Basic Input/Output System (BIOS) embedded on the motherboard and on other expansion cards, such as video adapters and Small Computer System Interface (SCSI) adapters, initiates the diagnostic tests. These tests appear on the monitor as vendor banners, memory counts, hard drive initializations, SCSI card and device enumerations, and listings for known ports, addresses, and **interrupt requests (IRQs)**. The POST also checks for the presence of floppy drives, a keyboard, and, in some cases, a mouse.
- Boot disk access: The bootstrapping process accesses the **master boot record (MBR)** on each hard drive. The MBR contains a copy of the partition table, which maintains or defines logical divisions on a hard drive. The partition table information is used to locate the active partition, also known as the **system partition**. The active partition contains the files used to load or initiate an operating system. In addition, the MBR defines which file (or files) should be loaded to initiate an operating system. After the computer hardware locates the active partition and the operating system loader, it transfers execution to that file (or files).

Note

On most systems, the complementary metal oxide semiconductor (CMOS) looks for a bootable floppy before starting from a hard drive. If a floppy is present, the computer attempts to start from that floppy. If an MBR is not present on the floppy, a "Non-system disk or disk error; Replace and press any key when ready" error occurs.

Next, the computer launches the operating system initialization files, thereby ending the bootstrapping process. Depending on the system configuration, the initialization files either launch a single operating system or enable the user to choose from several operating systems (known as a **multiboot system**). Windows 2000 supports multiboot capabilities with most other Microsoft operating systems without requiring additional software. A multiboot system that includes a non-Microsoft operating system, however, usually requires a partition management tool such as Partition Magic from PowerQuest (*http://www.powerquest.com/*) or System Commander from V Communications (*http://www.v-com.com/*).

Note

It is important to identify Microsoft-specific terminology when talking about operating system components. In Microsoft terms, the system partition contains the files used to initialize the Windows 2000 load process; the boot partition contains the main Windows 2000 directory and the pagefile. The system partition must be active. Although the boot partition can be the same partition as the system partition, it is typically located elsewhere on either the same or a different physical drive. Under Windows 2000, a boot partition can be transformed into a dynamic storage device, whereas a system partition must remain a standard basic storage device. On Alpha systems (which are RISC-based systems), a system partition must be formatted using file allocation table (FAT). On Intel systems, a system partition can be formatted with a FAT, FAT32, or New Technology File System (NTFS).

If no active partition exists on a hard drive, if the active partition is not the system partition, or if the files defined in the MBR are not present, any of several error messages can appear:

- Invalid partition table
- Error loading operating system
- Missing operating system

These error messages typically indicate that a functional operating system is not present on the computer. In some cases, they may signal that the hardware is not functioning properly or that the hard drive's partition table has failed or become corrupted. To fix these errors, you need to use some type of partitioning, configuration, or formatting tool. Luckily, the installation routines found on the four Windows 2000 setup boot disks include these functions.

Device Drivers

As mentioned in Chapter 2, a device driver is a small program designed to grant a specific operating system the ability to interact with and manipulate a device. An operating system requires device drivers to perform actions involving hardware access. Such actions include processing commands through the CPU, storing data in memory, reading input from a keyboard or mouse, reading data from a hard drive, creating a graphical display on a monitor, moving data across a network, and creating hard copy on a printer.

Windows 2000 includes most core device drivers (such as those for the CPU, memory, motherboard, PCI bus, and so forth) as part of the **hardware abstraction layer (HAL)**. Other hardware drivers, such as video system, audio, printer, network interface, and drive controller drivers, are easier to add, update, and change because they're not hardwired into (that is, permanent parts of) the HAL. After the initial installation of Windows 2000 is complete, you can modify or replace these drivers with newer, more powerful software.

Two types of device drivers operate in the Windows 2000 environment: **device class drivers** and **individual device drivers**.

Device class drivers supply basic driver interfaces and functions that define broad parameters for specific types of devices. Examples include Telephony Application Programming Interface (TAPI) modem drivers, graphic-rendering drivers within the printer subsystem, and

Integrated Drive Electronic (IDE) drivers for drive controllers. Individual device drivers are device- and model-specific programs that define the exact capabilities and functions of a particular device down to the make and model level; they allow the operating system to access the device's functions. These types of drivers are provided by the manufacturer whey you buy a new peripheral device, such as a printer, fax machine, or network interface card.

After a device driver is installed, it appears in the list of devices (the device and the driver are seen as the same object) within the Device Manager. The Device Manager (see Figure 3-2) is accessed either from the Hardware tab in the System applet (Start, Settings, Control Panel, System) or through the System Tools node in the Computer Management interface (Start, Programs, Administrative Tools, Computer Management).

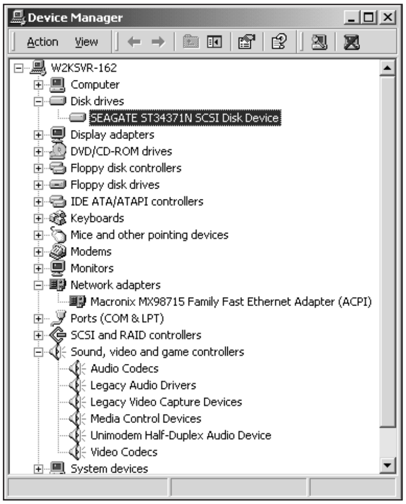

Figure 3-2 The Windows 2000 Device Manager

For more details on device drivers, see Chapter 5, "Device Management."

Base Operating System Loaders

After the computer identifies an active partition, the operating system launch procedure begins. This procedure varies with different operating systems, primarily because of the use of different filenames and the various tasks or functions assigned to files. Windows 2000 shares a common boot process and boot filenames with its predecessor, Windows NT. The boot process includes the following files, which are located in the root directory of the system partition (see Figure 3-3):

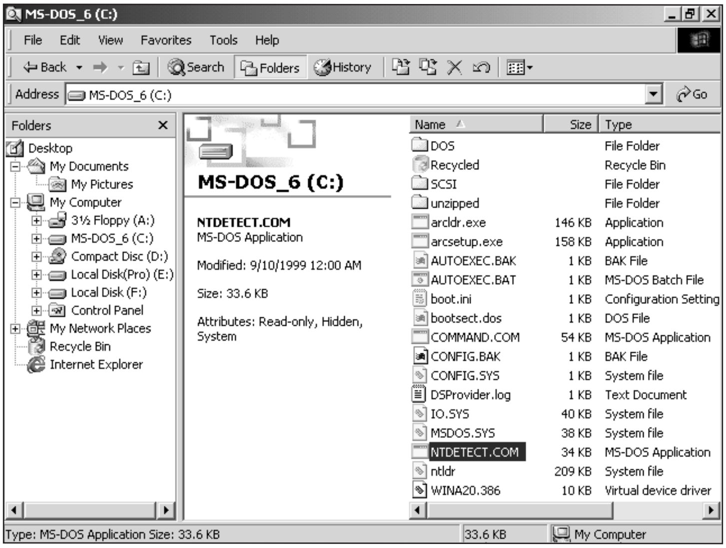

Figure 3-3 The files in the root directory of the system partition

- **Ntldr** is the operating system initialization file launched by the computer upon completion of the bootstrapping process. This file is responsible for loading Windows NT or other operating systems when they are present on a multiboot system. Ntldr uses the Boot.ini file to present a boot menu, which you use to select the operating system that should be launched.
- **Boot.ini** defines all relevant host partitions and the primary executables for those operating systems present on a computer. It also defines the default system to load when the boot menu display timer expires.
- **Ntdetect.com** executes before the Windows 2000 executable files are loaded from the boot partition. It performs hardware detection to create an inventory of devices and their configurations. A detected configuration is used to select a hardware profile, if more than one configuration is defined. The selected hardware profile determines which device drivers are loaded.
- **Ntbootdd.sys** appears only on systems with SCSI controllers that do not use an on-board BIOS translation. This file enables the drive controller system on the motherboard to control a SCSI adapter and its attached hard drives.
- **Bootsect.dos** is present only on a multiboot system where another Microsoft operating system, such as Windows 95/98, or a clone operating system, such as DOS, appears. This file establishes a boot environment more conducive to non-Windows 2000 and non-Windows NT operating systems.
- **Osloader.exe** appears only on Alpha systems. It replaces all of the preceding files that may be found on Intel machines by combining their functions into a single file. Alpha systems do not support multiboot systems.

When you select Windows 2000 from the boot menu or the boot menu timer expires with Windows 2000 as the default operating system, the Ntldr file executes Ntdetect.com. The latter file inspects the current state of hardware and attempts to match it with a known hardware profile. If the system cannot make a match, the user is prompted to select an alternative hardware profile. Hardware profiles are typically used on portable computers whose hardware devices change often, such as docking stations or network connections. After a hardware profile match is made, Ntldr launches **Ntoskrnl.exe** without releasing control of the system.

Ntoskrnl.exe is the Windows 2000 kernel, which is the core of the Windows 2000 operating environment. It controls the loading of all other files involved in establishing the computing environment. This file resides on the boot partition in the \Winnt\System32 folder (assuming you accept the default name for the system root during installation).

In addition to the Ntoskrnl.exe file, the boot process loads the Hal.dll file into memory. The Registry System key determines the system configuration and the order in which to load device drivers. Each driver required at start-up is loaded into memory (the progress of the loading process is indicated on the black screen by the display of several dots in the upper-left corner of the screen) and the Ntoskrnl. launches. When this operation is successful, the "Microsoft (R) Windows 2000 (TM) Server (Build 2072); 1 System Processor (128 MB Memory)" (or some equivalent) blue screen appears.

Next, the following system initialization functions occur:

1. All device drivers loaded into memory start.
2. Another Registry scan looks for the next set of device drivers to load (those required by the system but not required to initialize the kernel). These drivers are loaded and initialized.
3. The Session Manager (Smss.exe) launches to load and manage all services.
4. The BootExecute items launch. The Session Manager executes these commands before any services are loaded. BootExecute items include Autochk.exe (the Windows 2000 equivalent of the Windows NT Chkdsk.exe), which scans disks for problems or errors.
5. The Session Manager initiates the pagefile settings required by the Virtual Memory Manager.
6. The DOS Devices key is parsed to define all symbolic links to map to physical I/O ports, mail slots, named pipes, and so on.
7. The Windows 32-bit subsystem launches and initiates all I/O processes and the video display.

After all required drivers and files load and are initiated without significant problems, Winlogon.exe launches and starts the Local Security Administration (Lsass.exe). Next, the Begin Logon dialog box prompts the user to press Ctrl+Alt+Delete to log on. The start-up process is considered complete only after a successful logon.

THE WINDOWS REGISTRY

The **Registry** is a hierarchical database that stores the configuration and parameter details that govern Windows 2000 operation (see Figure 3-4). Windows 2000 also maintains text-based initialization files, such as Win.ini and System.ini, for backward compatibility, but these files are not required for Windows 2000 to run. Instead, the Registry handles the configuration and parameter details for the operating system and applications in the native Windows 2000 environment.

Caution

Microsoft recommends that you avoid direct interaction with the Registry whenever possible. In many situations, however, editing the Registry directly is the only possible way to solve a problem. Always back up the Registry and proceed with extreme caution whenever you find it necessary to modify the Windows Registry.

The Registry does not provide an exhaustive list of all applicable Windows 2000 configuration settings; rather, it lists only the exceptions to ordinary default settings for most entries. Defaults govern nearly all Windows 2000 behaviors unless a value in the Registry defines an alternative or additional setting. This approach creates a configuration environment in which some settings might not appear in the Registry; you must therefore change a built-in default manually. To make any changes to the Registry, you must be familiar with the exact syntax, spellings, path specifications, and valid values for Registry items. The Registry's structure is designed for system use and is not particularly user-friendly (as shown in Figure 3-4). The basic architecture of the Registry may be described as follows:

- The Registry is divided into five subtrees (listed on the left side of Figure 3-4).
- Each subtree is divided into keys.
- Each key is divided into subkeys.
- Any particular key can contain one or more layers or levels of subkeys. One or more value entries can occur within any key or subkey. A value entry includes a named parameter and one or more associated items of configuration data.
- Each value entry consists of three components: a name, a data type, and one or more actual data values.
- The data associated with any value entry describes its actual meaning or setting; the name identifies that setting; and the type indicates how the data entry should be interpreted.
- Registry data can be one of several types, including binary, decimal, text string, and hexadecimal.

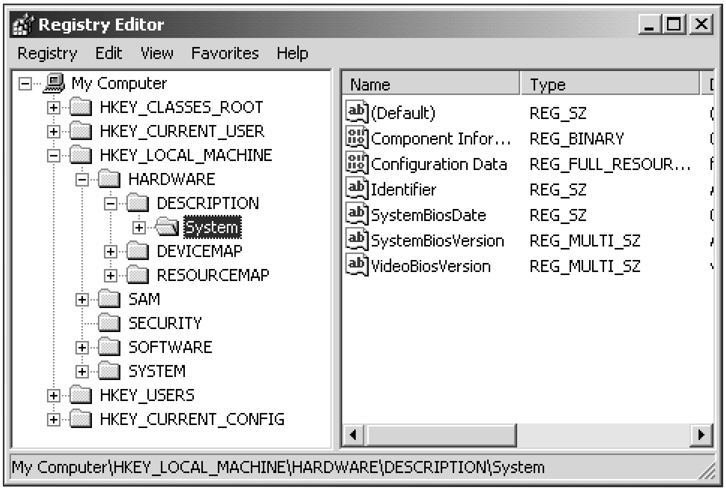

Figure 3-4 The Registry as seen through REGEDIT

The Windows 2000 Registry includes five subtrees:

- HKEY_LOCAL_MACHINE: This subtree includes the parameters that control the local computer system—specifically, hardware devices and device-dependent operating system components. The settings in this subtree are independent of users, applications, and processes.
- HKEY_CLASSES_ROOT: This subtree includes the parameters that control the relationships between file format types and applications. This relationship is based on the file extension (the characters to the right of the last period in a filename, such as "doc" in "Readme.doc"). The HKEY_CLASSES_ROOT subtree also maintains information about object linking and embedding, COM objects, and file-class associations.
- HKEY_CURRENT_CONFIG: This subtree contains the parameters that control the environment for the currently active hardware profile. Its contents are built or copied each time the system is started just after a hardware profile is selected. The contents are copied from the appropriate subtree of the HKEY_LOCAL _MACHINE\ System\CurrentControlSet\HardwareProfiles\ subtree. The HKEY_CURRENT_CONFIG subtree is actually a compromise intended to maintain compatibility with Windows 95 applications that reference this subtree, instead of the HKEY_LOCAL_MACHINE subtree, so as to obtain configuration information.
- HKEY_CURRENT_USER: This subtree includes the parameters that control the user environment for whichever user account is currently logged into a Windows 2000 system. This subkey is loaded into memory from the Ntuser.dat (or Ntuser.man) file found in a user's profile. The most recent copy of the user's profile information is copied from HKEY_USERS key, if that user's profile data is present; otherwise, the default user profile is loaded.

- HKEY_USERS: This subtree includes profile information settings for all users who have logged into this computer, along with the default user profile.

The Registry items that actually define or change the way in which objects, services, and functions operate are found in the Registry value entries. As noted earlier, each value entry has three components: name, data type, and value. A value entry name often consists of a capitalized phrase with no embedded spaces, such as DefaultUserName. The data contained in a value entry is stored in one of five primary data types:

- REG_BINARY: Binary format for arbitrarily large amounts of Registry data.
- REG_DWORD: Binary, hexadecimal, or decimal format for a maximum of 32 bits of Registry data.
- REG_SZ: A Unicode text string format for Registry data.
- REG_MULTI_SZ: A Unicode text string containing multiple values separated by NULL characters.
- REG_EXPAND_SZ: An expandable text string containing variables (such as %systemroot%) to be replaced by an application.

You must explicitly define the data type used to store data when creating a value entry. It is not possible to transform a value entry from one data type to another. To change the data type of a value entry, you must delete the entry and re-create it using a new data type.

The term hive often crops up in discussions of the Registry. A **hive** is a section of the Registry that is stored in a separate file. Hives are permanent structures; they are saved each time that the system shuts down and reloaded each time the system powers up. Hives are not rebuilt or constructed each time that Windows 2000 boots. These sections are stored in one of two locations: %systemroot%\System32\Config (all hives) and \Documents and Settings\%username% (only Ntuser.dat, or Ntuser.man). Table 3-1 lists the hives stored by Windows 2000 and their related filenames.

Table 3-1 Filenames of Registry hives

Registry Hive	Filenames
HKEY_LOCAL_MACHINE\SAM	Sam, Sam.log, Sam.sav
HKEY_LOCAL_MACHINE\Security	Security, Security.log, Security.sav
HKEY_LOCAL_MACHINE\Software	Software, Software.log, Software.sav
HKEY_LOCAL_MACHINE\System	System, System.alt, System.log, System.sav
HKEY_CURRENT_CONFIG	System, System.alt, System.log, System.sav
HKEY_USERS\.DEFAULT	Default, Default.log, Default.sav
(Not associated with a hive)	Userdiff, Userdiff.log
HKEY_CURRENT_USER	Ntuser.dat, Ntuser.dat.log

As you can see from Table 3-1, several types of files are used to store the Registry hives. The four main file types are as follows:

- No extension indicates the hive file itself.
- .alt indicates a back-up copy of the hive file. Only the System hive has an .alt file.
- .log indicates a log of all transactions or changes made to a hive.
- .sav indicates a copy of the hive file created at the end of the text portion of the initial setup process.

The Registry layout, structure, and files are complex. Nevertheless, the design of the Registry makes it easy for the operating system to use the Registry from memory and to provide basic fault tolerance. Each time the system starts, the entire Registry is loaded into memory; while the system is running, however, only this memory-resident version of the Registry is used. Changes to the Registry are made to the memory-resident version and become effective immediately (when the changes affect a core process, the system must be restarted). The transaction logs record all changes made to the memory-resident Registry. When the system shuts down, the transaction log is used to implement the same net changes to the hive file (the hive file is not copied from memory to the hard drive). When the system is started again, the log file is used to verify that the changes it recorded were made into the hive file: the log file is then cleared.

Because the System hive is extremely important, changes are made to its main hive file, then to its .alt file. This approach ensures that a functioning hive file remains available even if the change process is interrupted.

Note

The Windows 2000 Registry has only five default subtrees (described in the following sections). Installing some Windows 95 or Windows 98 software can result in the creation of a sixth subtree, HKEY_DYN_DATA. This subtree does not really exist. Instead, Windows 2000 creates a virtual link to other areas of the Registry that already support the data structure represented by this key in Windows 95/98. This process maintains compatibility for older Win32 applications.

HKEY_LOCAL_MACHINE

The HKEY_LOCAL_MACHINE subtree, shown in Figure 3-5, contains device configuration data. This information is independent of users, applications, and processes. Applications, device drivers, and kernel services use this data to load and configure device-dependent functions, objects, and services. The key is divided into five subkeys: Hardware, SAM, Security, Software, and System.

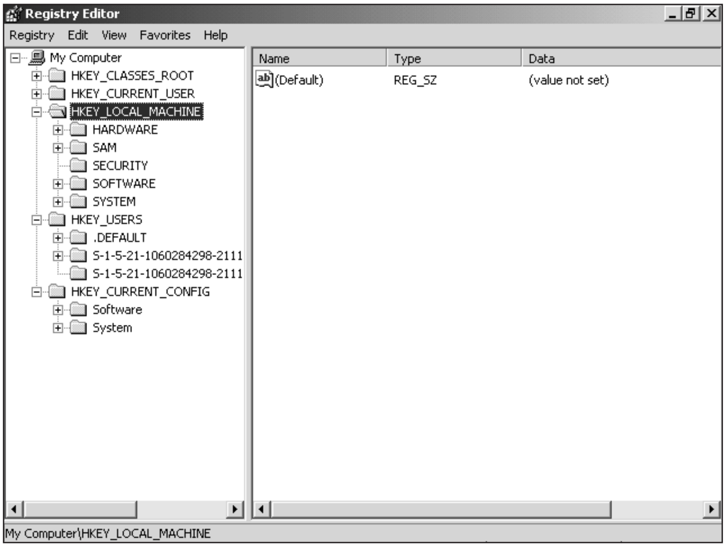

Figure 3-5 The HKEY_LOCAL_MACHINE subtree

HKEY_LOCAL_MACHINE\Hardware

The Hardware subkey of HKEY_LOCAL_MACHINE stores data specific to current hardware attached to the local computer, including physical settings, I/O ports, memory requirements, driver mappings, and IRQ settings. The data in this subkey is regenerated by the Ntdetect.com utility each time that the system starts and is not saved to a hive file whenever a shutdown occurs.

HKEY_LOCAL_MACHINE\SAM

The Security Accounts Manager (SAM) subkey of HKEY_LOCAL_MACHINE stores security-related data, including the SAM database and information about user accounts and groups. Most experts recommend that you never edit the data in this subkey directly. Use the Windows 2000 Active Directory, accounts, and group management tools to manage this sensitive and important system data instead.

HKEY_LOCAL_MACHINE\Security

The Security subkey of HKEY_LOCAL_MACHINE stores data specific to security policies (that is, group policies and local security policies). Most experts recommend that you never edit the data in this subkey directly; use the Windows 2000 Policy Editor and related tools to manage this sensitive system data instead.

HKEY_LOCAL_MACHINE\Software

The Software subkey of HKEY_LOCAL_MACHINE stores data related to installed software components. It is usually subdivided into subkeys, with one subkey for each software

vendor with one or more products installed. For example, all Microsoft software information is stored in the HKEY_LOCAL_MACHINE\Software\Microsoft subkey.

HKEY_LOCAL_MACHINE\System

The System subkey of HKEY_LOCAL_MACHINE stores data related to the Windows 2000 start-up process, including start-up parameters, initialization instructions for device drivers, service parameters, and descriptions of basic operating system behaviors. The data in the System subkey are divided into control sets. A **control set** is a hardware profile-specific collection of start-up process parameters. The CurrentControlSet is a link to the ControlSet### subkey that contains data that describe the current active session. The numbered subkeys that appear beneath the System subkey, such as ControlSet001 and ControlSet002, represent unique hardware profiles and backups of previous successful system starts.

HKEY_CLASSES_ROOT

The HKEY_CLASSES_ROOT subtree, shown in Figure 3-6, contains mappings between application file extensions and data essential to COM objects and maintains a copy of the information in the HKEY_LOCAL_MACHINE\Software\Classes subtree. This subtree provides backward compatibility with Windows 95/98 applications.

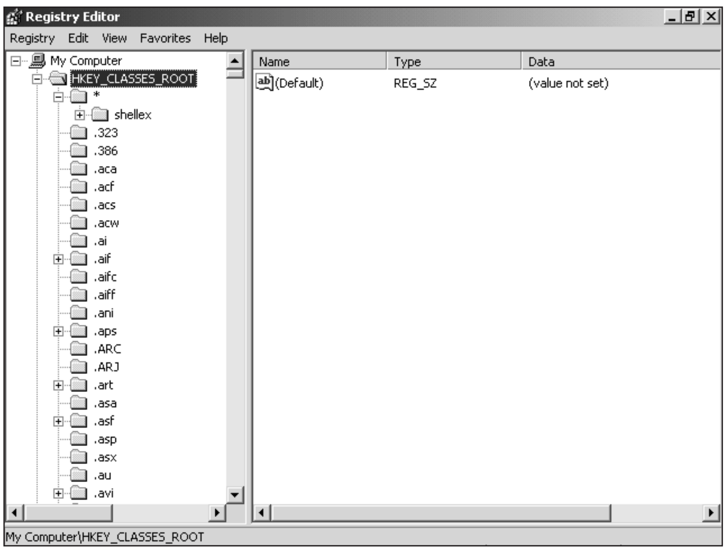

Figure 3-6 The HKEY_CLASSES_ROOT subtree

HKEY_CURRENT_CONFIG

The HKEY_CURRENT_CONFIG subtree, shown in Figure 3-7, maps to the current control set used by the system, which appears in HKEY_LOCAL_MACHINE\System\CurrentControlSet\Hardware Profiles\Current. This subtree supports backward compatibility with Windows 95/98 applications.

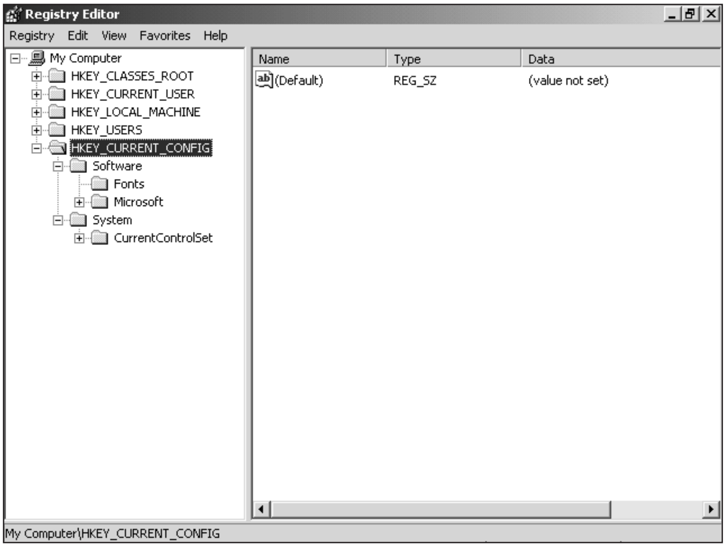

Figure 3-7 The HKEY_CURRENT_CONFIG subtree

HKEY_CURRENT_USER

The HKEY_ CURRENT_USER subtree, shown in Figure 3-8, contains the current user's profile and is created whenever a user logs onto a Windows 2000 machine. It is created by copying profile data from HKEY_USERS and loading the Ntuser.dat file, or by duplicating the default user profile from HKEY_USERS if no profile for that user appears in Ntuser.dat.

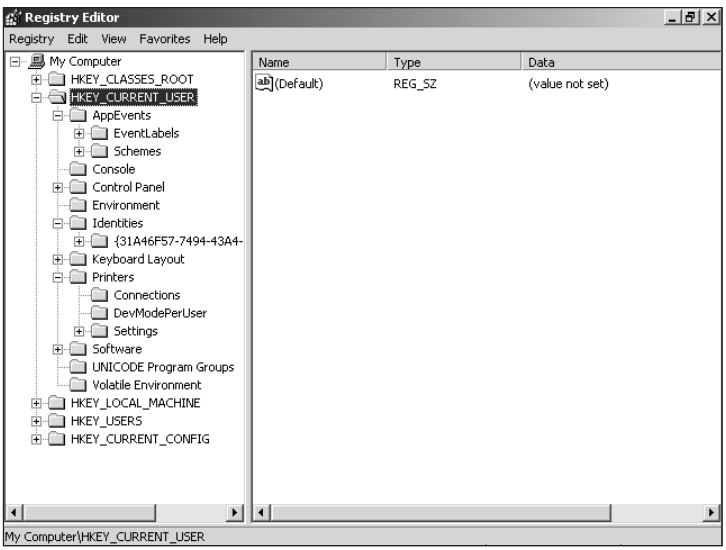

Figure 3-8 The HKEY_CURRENT_USER subtree

HKEY_USERS

The HKEY_CLASSES_ROOT subtree, shown in Figure 3-9, contains user profiles for all users who have logged onto this computer at any time in the past. It also includes a default user profile that is invoked whenever an existing user profile is not available.

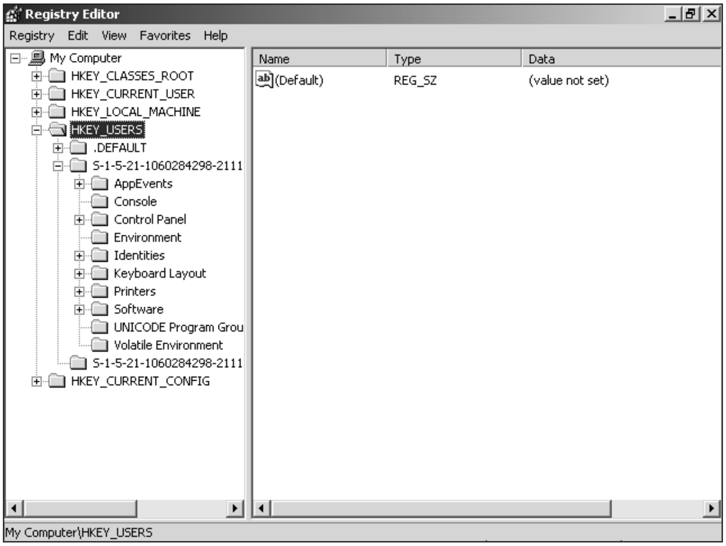

Figure 3-9 The HKEY_USERS subtree

Protecting the Registry

Protecting the Windows 2000 Registry involves following two essential rules. First and foremost, avoid editing the Registry directly. Instead, use an interface such as an administrative tool, a Microsoft Management Console (MMC) snap-in, or some other utility that edits the Registry.

Second, back up the Registry whenever it is altered. You can then restore the back-up copy if the original becomes damaged or corrupted.

Backing Up the Registry

You can back up the Registry in any one of the following ways:

- Use back-up software designed for Windows 2000 that includes Registry backups with full system data backups.
- Use Microsoft's REGEDIT or REGEDT32 tools to create Registry backups manually (see Hands-on Project 3-6 to see how this backup is done).
- Use a Support Tools utility designed to back up the Registry (REG COPY, REG SAVE, or REG BACKUP).

Note

The emergency repair disk (ERD) that Windows 2000 creates as a system repair tool no longer contains a copy of the Registry. For this reason, the ERD is not considered a valid way of maintaining even a partial Registry backup.

If a Windows 2000 Registry becomes corrupted, you might be able to restore the system to normal activity through one of several repair maneuvers. The first such maneuver involves the Last Known Good Configuration start-up option, which attempts to restore the Registry to its (presumably working) state at the time of the last successful logon. The second maneuver is to use the Windows 2000 Safe Mode boot option, which boots the system using only basic drivers and required system files. The third maneuver is to restore all or part of the Registry from a backup. Try these repair strategies in the order they are presented here, because the amount of work and time required to complete each one increases from first to last. Refer to Chapter 15, "Troubleshooting Windows 2000," for more information on how to perform these procedures.

Modifying the Registry

If you must edit the Windows 2000 Registry directly, you can use one of two editors: REGEDIT and REGEDT32. You can launch these tools manually from a command prompt or by using the Run command. REGEDIT is a 16-bit application, and REGEDT32 is a 32-bit application.

REGEDIT, which originally appeared in Windows 95, displays all Registry subtrees in a single window and allows exhaustive searching of the entire Registry's contents. In contrast, REGEDT32 displays each subtree in its own cascading window. Its primary benefit is improved controls for security settings and the ability to audit Registry access on a subtree, key, subkey, or value entry basis.

Either Registry editor can be used to add, modify, and delete value entries and subtrees, import and export all or part of the Registry, and access and edit Registries on remote systems. When using either Registry editor, however, you risk damaging the Registry. Because the Registry is the primary control mechanism for Windows 2000, it is possible to destroy an installation because of a minor editing mistake. Whenever you edit the Registry directly, remember to do the following:

- Create a back-up of all Registry keys to be edited.
- If possible, back up all data on another computer to prevent loss of access to back-up data if the system fails.
- Perform only a single change at a time.
- Reboot the system between changes, even if not prompted to do so.
- Test changes on nonproduction systems before altering a mission-critical system or deploying wholesale changes on numerous systems.

- Use REGEDT32 in read-only mode when you are exploring the Registry to prevent accidental changes or damages.

For more information on the Registry, consult the Windows 2000 Support Tools. For details on individual value entries, consult the REGENTRY tool on the Windows 2000 installation CD in the Support\Tools directory.

BASE SYSTEM FUNCTIONALITY

As discussed in Chapter 1, any operating system must support some essential operations. In the following sections, these basic operations are covered in more detail as they relate to Windows 2000.

Writing to Disk

Writing to disk is a key function that allows data to be retained even when a computer is not running. In this process, information that is stored in memory becomes recorded on a physical disk drive or some other persistent storage medium (such as a CD-ROM or tape drive). Data stored in memory is considered transitory, because all data in memory is lost when power is not available. Data stored on a physical drive is considered permanent, because the information is retained even when power is no longer available.

The Windows 2000 system writes data, including page swapping, Registry changes, and user profile changes, to disk for several reasons. Most often, disk writing activities are initiated by user applications to record data relevant to the application in the form of one or more specially formatted data files. For Windows 2000, writing to disk involves the File Systems Manager module in the **executive services** portion of the kernel, because such activity usually requires accessing one or more files. The following executive services modules can be involved as well:

- The Object Manager handles the initial request for which file object (or objects) may be requested for access.
- The Security Reference Monitor is called, because the Object Manager inquires whether access to the requested object and related service may proceed.
- The I/O Manager handles access to the device class driver and the actual device driver for the physical drive that is accessed, if the Object Manager calls it to do so. (By implication, the Security Reference Monitor must have granted access to the object for whatever service is requested.)

When virtual memory is accessed, the Virtual Memory Manager may instigate writing to the disk as well, if the operating system needs to read from or write to the **paging file** stored on disk. (A paging file—also known as a pagefile—is temporary storage space on a hard drive.)

Allocating Memory

As discussed in Chapter 2, memory allocation is the process whereby sections of memory are assigned to a process or device driver for its exclusive use. This function is important; without

it, processes and device drivers might interfere with each other's operations by overwriting data stored in memory.

Windows 2000 employs **virtual machines** whenever possible, even on systems with ample amounts of physical RAM. Each virtual machine can access as much as 4 GB of address space. The Windows 2000 executive services Virtual Memory Manager juggles address spaces for all virtual machines, mapping all used address spaces to actual memory pages. The Virtual Memory Manager allocates memory dynamically, based on availability and use. When a process requires such a memory page, the page is loaded into physical RAM (if it is not already present). When a page is not in use, it can be pushed out to the pagefile to release physical RAM for other uses, such as other processes or threads that are active on the same machine at roughly the same time.

On the one hand, some device drivers—specifically, real-mode device drivers—require fixed, predetermined physical memory allocations to work. On the other hand, protected-mode device drivers can reside in areas of memory that might not always be present in physical RAM. Fortunately, the Virtual Memory Manager is aware of those memory areas that real-mode drivers must use exclusively and will not page those sections to disk or map them into virtual machine address spaces.

Allocating CPU Time

Allocating CPU time means controlling when and how much execution time is granted to processes. Most modern operating systems grant access to the CPU for a maximum fixed time interval known as a time slice (this operation is sometimes called time slicing). CPU allocation is important in a multithreaded operating system because it prevents any single thread from dominating access to the CPU. By assigning explicit execution priorities to processes and by limiting the maximum amount of time that the process is allowed to access the CPU before stepping aside for another process, more than one process can appear to be active simultaneously, even on a single-CPU system. That's because humans are so much slower than computers. Although a process might be able to accomplish significant work before its time slice expires and other processes become active, to ordinary mortals all processes that are sharing the CPU appear to be active at the same time.

Windows 2000 uses a scheduling algorithm for the CPU called **preemptive multitasking**. In plain English, even though a process may expect to occupy the CPU for an entire time slice, once it becomes active, it can be preempted at any time when a higher-priority task requires service. This task-switching capability lets Windows 2000 maintain the operating system kernel, control hardware access, sustain virtual machines, and allow user-mode processes to execute, all apparently simultaneously. In reality, of course, the operating system executive is scheduling processes, monitoring their progress, and switching them out continuously, depending on elapsed time or the arrival of higher-priority processes or tasks.

Windows 2000 also supports a different scheduling regimen called cooperative multitasking for Windows 16-bit applications. The original environment of Windows 16-bit applications allowed processes to retain access to the CPU until those processes voluntarily relinquished

such access. Thus those processes were designed to surrender possession of the CPU on a cooperative basis to permit other tasks to execute.

Because cooperative multitasking is fundamentally incompatible with preemptive multitasking, Windows 2000 provides a runtime environment that emulates cooperative multitasking within the context of a single virtual machine. By default, Win16 applications operate within the same virtual machine. Within that virtual machine, the Win16 processes follow the cooperative multitasking native to such applications, which allows Windows 2000 to support what appears to be cooperative multitasking in a preemptive multitasking environment. In reality, the single virtual machine in which Win16 applications execute is subject to the time slice limitations, execution priority settings, and possibility of preemption by other processes, just as all other Windows 2000 virtual machines are.

The allocation of CPU time is managed primarily by the scheduling and priority management aspects of the operating system kernel, which is always resident in memory. Although the Process Manager is also involved, it is responsible for maintaining active virtual machines, not allocating CPU access.

Handling Data-Processing Requests

Handling data-processing requests basically means responding to resource requests that originate in a particular process that is active at the time the request is made. The operating system must direct that request to the proper provider, possibly instructing that provider to respond to the request. For example, when an application requests a file, the request is captured by the virtual machine, routed to the kernel's executive services, and processed by the Security Reference Monitor, the Object Manager, and the File Systems Manager. Ultimately, that request is routed to the drive where the file resides through the agencies of the I/O Manager, and its contents are obtained. This information then follows the same trail in reverse, going back to the requesting process.

Resource requests can include requests for memory, CPU access, files, graphical objects or information, and keyboard input or output. Without the ability to handle such requests, any operating system, including Windows 2000, is useless. This point explains the fundamental impetus for the Windows 2000 architecture, especially the executive services components and their modular design. Each of those modules is designed to handle particular kinds of resource requests or to expedite the process of handling resource requests in general. Their modularity helps ensure that individual components can change to improve their capabilities or performance without materially affecting other aspects of the system. In this respect, Windows 2000 resembles a complex machine made of interlocking parts, all of which are necessary for the operating system to do its job.

Error Handling

Error handling involves recognizing and averting the effects of invalid data or attempts to execute illegal actions before they can damage a system or the resources managed by that system. It also requires generating a system response—perhaps an error message to users and

some kind of post-error cleanup behavior. In turn, this step requires dealing with issues specific to the context within which errors occur and responding with the appropriate action.

Error handling is primarily a defensive reaction to an anomalous system or application condition. From an operating system perspective, error handling applies to its kernel, subsystem modules, and utilities and device drivers. At this level, the primary function of error handling is to prevent system crashes by redirecting problems to an error mechanism rather than attempting to continue processes or operations with incorrect data or to execute illegal or invalid instructions.

Within Windows 2000, errors are handled by suspending the process associated with the error in noncore system code. Because processes remain isolated from one another, the failure or suspension of a single process should not cause the entire operating system to crash. If an error can be resolved, the suspended process can be released to continue functioning. If it cannot be resolved, the offending process is usually terminated. The worst-case scenario for an application error is that the application in which the error occurred terminates, losing whatever data that has not yet been saved to disk.

On the other hand, if an error occurs in some component of the operating system itself, a so-called STOP error occurs. (Because these types of errors are reported in a special format on an all-blue, character-only screen, they are called the "blue screen of death," abbreviated as BSOD.) STOP errors cause the operating system to halt and cease its operations. It is possible to configure STOP error management to record details about whatever event triggered the error, to restart the system, and even to write the current contents of memory to a dump file for debugging inspection by a system professional.

Network Functionality

Network functionality is the ability to communicate with other computers across some kind of communications medium. This capability relies on a shared protocol (communication language), a shared physical medium (cable, telephone wire, infrared, microwaves, and so on), and a network interface (often an expansion card or PC Card with its related device drivers).

Windows 2000 includes network functionality as a basic element of its design. The ability to communicate over a network permits the sharing of resources and distribution of the workload. In addition, it simplifies interpersonal communications, promotes centralized administration and control, and offers a broader range of access to resources—from across the room to across the world. With the phenomenal rise of the Internet, no operating system released in the latter part of the 1990s can be considered complete if it does not offer built-in networking support. In fact, the last operating system from Microsoft that failed to include built-in networking functionality was Windows 3.1 (released in 1990).

Interprocess Communications (IPCs, RPCs)

In the Microsoft environment, interprocess communication (IPC) employs a proprietary but well-documented API that processes use to exchange information. In more general terms, IPC defines a mechanism that allows one active process to exchange data with another

process. If both processes are active, information can move more or less directly from sender to receiver. If the receiving process is unavailable or inactive, however, an intermediary is required to store the information for delivery to the receiver if and when the receiver becomes active. Handling these kinds of complexities is what makes IPC so interesting and the software that implements IPC so challenging to create.

In the Windows 2000 environment, IPC allows programmers to employ well-defined query and response mechanisms without requiring them to develop these mechanisms themselves. IPC enables the kernel, executive services, virtual machines, and user processes to function as they do—that is, to communicate with each other so as to direct activity within the operating system.

Remote process communication (RPC) is similar to IPC; it allows processes running on different machines to communicate. RPC occurs across local networks or wide area network (WAN) links. This fundamental technology enables client-server applications to operate and distributed computing to occur.

Messaging

In general, messaging allows operating systems to communicate across a network or over a remote access link. It relies on a server service that listens on a specific port address to which clients direct resource requests across the network.

Messaging is a generalized concept that includes both IPC and RPC mechanisms. In Windows operating systems, messaging also includes esoteric mechanisms such as named pipes, mailslots, and NetBIOS interfaces. Windows 2000 takes advantage of this capability, especially with powerful clients. When either Windows 2000 Professional or Windows NT 4.0 Workstation operates as a client, the messaging features of Windows 2000 support a much broader range of interaction, control, and resource access.

Terminal Services

Terminal Services refers to a class of computer services that traditionally describes a mechanism that permits a so-called **smart terminal**—not much more than a monitor and a keyboard with a network attachment—to access and use the services of an operating system across a network.

In their earliest manifestations, terminals defined a typical user interface for mainframes and minicomputers. In such environments, the computer was a large, exotic, and expensive collection of equipment kept behind locked doors and maintained to by a specially trained staff. Ordinary users could access the computer only through a terminal, which gave them the ability to input data and queries and to view the results of their work, but included little or no additional computing abilities.

Today, two distinct categories of Terminal Services persist: client emulation and remote control. Although it remains possible to use Terminal Services with nothing more than a smart terminal and keyboard, client emulation allows PCs and other stand-alone computers to act as if they were nothing more than smart terminals from the serving computer's point of view.

Windows 2000 includes an add-on that supports Terminal Services, and both types of clients (smart terminals and PCs that are emulating smart terminals) can take advantage of its services.

Remote control, on the other hand, gives one operating system the ability to control a remote system or to use that remote system's full client capabilities as if it were physically present. This feature simplifies administration by allowing a single client to control systems throughout a network. In either case, the remote control aspect of Terminal Services allows an entire desktop environment to be maintained in a virtual machine on a server that any client can access across the network (even if that client is not a Windows machine).

Microsoft designed its Terminal Services with a particular class of desktop computers in mind—**thin clients**, also known as **network computers**. Both terms refer to a configuration similar to the original smart terminal in the mainframe and minicomputer eras: a low-cost, low-powered desktop environment. For the more modern implementations, this configuration consists of a networked computer with just enough CPU power and memory to handle local input and output tasks. This client machine relies on a server elsewhere on the network to run applications, provide storage, and supply the many services that workplace users expect today, such as e-mail, Internet access, and productivity applications.

Terminal Services allow inexpensive client devices to deliver all of the high-powered, complex services and applications that can run on today's high-performance servers. It also allows organizations to control client desktops much more rigorously and makes these desktops much easier to maintain and update in a rapidly changing technological landscape. Although it's not clear whether this kind of computing will ever rule the world again, as it did in the 1960s when mainframes and minicomputers were the norm, it does offer an attractive option for some organizations. Clearly, this kind of computing works better than using stand-alone PCs (primarily for cost and control reasons) for certain applications—such as point-of-sale devices ("smart" cash registers), factory floor monitors and devices, and loading dock systems. Such computing is expected to remain in active use for the foreseeable future.

Microsoft apparently agrees with this analysis, because Windows 2000 Server includes Terminal Services as an add-on element. This feature allows lower-end PCs or thin clients to participate in the rich environment of Windows 2000 without meeting the hardware requirements needed for installing Windows 2000 Professional on those client machines. Terminals or thin clients with access to Terminal Services need little more than a bootstrap mechanism to access the network and a Windows 2000 Server running the right software. After they're connected, the Terminal Services process maintains the user environment, transmitting only the graphical display information and receiving only the input from the keyboard or mouse that the client needs to manage local input and display information.

TOOLS FOUND IN WINDOWS 2000 SERVER

Windows 2000 Server offers a wide variety of system control, administration, and management tools. These tools allow you to fine-tune, control, and manipulate all aspects of the operating system. In an effort to standardize control interfaces, Microsoft introduced the Microsoft

Management Console (MMC) in Internet Information Server 4.0. With the release of Windows 2000, the MMC becomes the operating system's standard management interface.

The MMC is a standardized graphical interface into which control modules called snap-ins can be loaded to provide administration and control functionality over specific types or groups of objects within the Windows 2000 environment. The MMC provides no control functions itself; rather, it merely defines an environment and a set of well-documented mechanisms to control underlying systems, related subsystems, and operating system objects of many types.

In fact, the Windows 2000 administration and control tools combine MMC snap-ins, multitabbed dialog boxes, and step-by-step software wizards. You can access all of these control utilities through the Windows 2000 Control Panel. Some of the more important tools in this collection include the following:

- Add/Remove Hardware: Use this wizard to install drivers for new hardware, remove drivers for hardware to be removed from the system, and troubleshoot malfunctioning hardware.
- Add/Remove Programs: Use this tool to change or remove installed applications, setup new applications, and alter the installed components of Windows 2000.
- Network and Dial-Up Connections: Use this tool to configure all network communications, including settings for NICs, services, and protocols. This interface manages both direct cable connections, such as those used in a LAN, and remote links, such as those created over telephone or ISDN lines.
- System: Use this multitabbed dialog box to change network membership and computer name, access the Device Manager, define hardware profiles, create roaming user profiles, define performance options, set pagefile settings, define environmental variables, and define start-up and recovery options.
- Active Directory Users and Computers: Use this tool to define and manage user accounts and groups. You access this tool via the Start menu or the Administrative Tools icon in the Control Panel.
- Computer Management: Use this tool to access the Event Viewer, obtain system information, review performance, manage shared folders, access the Device Manager, manage physical disks, and manage system services, such as Dynamic Host Configuration Protocol (DHCP), telephony, Internet Information Services (IIS), and Domain Name Service (DNS). Access it via the Start menu or the Administrative Tools icon in the Control Panel.
- Distributed File System: Use this tool to create a single-access-point, single-hierarchy system for a shared resource on a network when the resources are hosted by multiple servers. Access it via the Start menu or the Administrative Tools icon in the Control Panel.
- Routing and Remote Access: Use this tool to manage multihomed server-based routing and remote access to and from the server. Access it via the Start menu or the Administrative Tools icon in the Control Panel.

Many of the tools found in Windows 2000 are MMC snap-in transformations of previous controls found in Windows NT or tools migrated and improved from Windows 98. The result is a set of powerful, easy-to-use tools that manage and control every aspect of the Windows 2000 environment.

CHAPTER SUMMARY

- An operating system consists of a collection of files, such as Ntoskrnl.exe, DLLs, and device drivers. Specific files are required during start-up to establish the environment so that the operating system as a whole can be loaded and maintained.
- Windows 2000 employs a modular approach in its system architecture that results in a stable and functional system. The major functions and capabilities of Windows 2000 are delegated to and managed by the executive services. The Registry contains configuration and operational settings that Windows 2000 uses to control its environment.
- Windows 2000 demonstrates all of the basic system functionalities that a modern operating system must support. Various administration and management utilities control the environment of Windows 2000, primarily those found in the Control Panel and under the Administrative Tools menu.

KEY TERMS

Boot.ini — A file that defines the host partitions and the primary executables of the operating systems present on the computer. Boot.ini also defines the default operating system that loads when the customizable boot menu display timer expires.

boot partition — On a Windows 2000 system, the partition that contains the main operating system directory and the pagefile. The boot partition can be the same as the system partition, but most often is located elsewhere, either on the same drive or on a different physical drive.

Bootsect.dos — On a multiboot system with another Microsoft, clone, or near-equivalent operating system such as DOS or Windows 95/98, a file that is used to establish a start-up environment more conducive to these older Microsoft operating systems.

bootstrapping — The initialization process that a computer goes through to inspect its hardware and locate the boot files for an operating system on the active partition of a hard drive.

bootstrapping files — Computer files required to initiate loading and launching an operating system. Also called boot files.

control set — A hardware-profile-specific collection of boot process parameters.

core system files — Those files that make up the core components of an operating system. If these files become corrupted or damaged, the operating system cannot function.

device class driver — A piece of software that supplies basic driver interfaces and functions that define broad parameters for specific types of devices.

executive services — The collection of all intermediary and management components for all resources, security, and communications in the Windows 2000 environment.

User-mode processes do not actually interact with executive services; rather, they interact with APIs defined for their application subsystems. The virtual machine in which the calling application runs then redirects such API calls to the kernel, where they are routed to the appropriate executive service.

hardware abstraction layer (HAL) — The only module of Windows 2000 that is hardware-specific. The HAL is built to match the type and state of the hardware during installation.

hive — A section of the Registry that is stored in a separate file. Hives are permanent structures that are saved each time the system is shut down, and reloaded each time the system is powered up.

individual device driver — A device- and model-specific program that defines the exact capabilities and functions of a particular device down to the make and model level, and allows the operating system to access the device's functions.

interrupt request (IRQ) — A special, high-priority communications channel through which a hardware device informs the CPU that it needs to perform some action or respond to some condition.

master boot record (MBR) — The section on a hard drive where the partition table and other key descriptive information are stored.

modular architecture — A method of programming where multiple separate components are combined into a single logical whole. Each component handles a specific task or a small set of related tasks. Windows 2000 uses such architecture in its kernel mode, particularly for the components that make up its executive services.

multiboot system — A computer that contains two or more operating systems and allows the user to select which operating system to start during each initial system start-up cycle.

network computers — See thin clients.

Ntbootdd.sys — A file that appears on Windows 2000 and Windows NT systems with SCSI controllers that do not have an on-board BIOS translation enabled or present. It enables the drive controller system on the motherboard to control a SCSI adapter and its attached hard drives.

Ntdetect.com — A file that is invoked just prior to loading the Windows 2000 executable files from the boot partition. It performs a hardware inspection to create an inventory of devices and their configurations. The configuration that is detected is used to select a hardware profile, which in turn determines which device drivers are loaded.

Ntldr — The operating system initialization file that the computer launches upon the completion of the bootstrapping process. It is responsible for loading Windows NT or other operating systems when it appears on a multiboot system. Ntldr uses the Boot.ini file to present a boot menu, which in turn is used to select the operating system to be launched.

Ntoskrnl.exe — A file that contains the Windows 2000 kernel, which is the core of the Windows 2000 operating environment. It controls the loading of all other files involved in establishing the computing environment. Ntoskrnl.exe resides on the boot partition in the \Winnt\System32 folder (assuming the default name for the system root is accepted during installation).

Osloader.exe — A file that appears only on Alpha systems. It replaces all of the various files found on Intel machines by combining their functions into a single file.

paging file — See pagefile.

pagefile — Temporary storage space on a hard drive.

Power On, Self-Test (POST) — An internal diagnostic that a computer performs during the earliest phases of the bootstrapping process.

preemptive multitasking — Type of multitasking in which the memory manager controls who has control of the CPU, rather than giving the responsibilities to the applications.

Registry — The configuration and parameter details that govern how Windows 2000 operates. The Registry is stored as a single hierarchical database.

smart terminal — A computer that has only a monitor and a keyboard with a network attachment.

system partition — The partition that contains the files used to initialize the Windows 2000 loading process.

thin clients — A low-cost, low-powered desktop environment with just enough CPU power and memory to handle local input and output tasks.

virtual machine — A software construct that creates a computer environment for each process, so that the process appears to be the exclusive resident of the physical machine. In Windows 2000, application subsystems construct virtual machines for processes. When a process requests access to a resource (whether memory, CPU time, keyboard input, display changes, or hard drive resources), the virtual machine relays that request to the application subsystem in which the virtual machine resides. This subsystem, in turn, passes the request to the appropriate executive service in the kernel mode.

Review Questions

1. Which of the following statements are true? (Choose all that apply.)
   a. MS-DOS requires several specialized files to load itself as an operating system.
   b. Windows 2000 is a monolithic code block.
   c. Modular architecture improves fault tolerance.
   d. Individual modules of Windows 2000 cannot be replaced.
2. What happens immediately after a computer's power is turned on?
   a. The Registry is loaded.
   b. Boot.ini is read.
   c. POST takes place.
   d. Ntoskrnl.exe is launched.
3. Which of the following actually determines from where the computer attempts to load an operating system?
   a. Active partition
   b. Partition table
   c. Registry
   d. File system

4. Windows 2000 cannot exist on a computer in a multiboot configuration. True or False?
5. In the Windows 2000 context, the active partition is also the ________________ partition.
6. On Alpha systems hosting Windows 2000, the system partition must be formatted with which of the following?
   a. FAT
   b. FAT32
   c. NTFS
   d. HPFS
7. What tool can you use to partition a hard drive?
   a. Registry
   b. Setup Boot disks
   c. MBR
   d. Boot Manager
8. Windows 2000 and Windows NT have nearly identical boot methods. True or False?
9. Which Windows 2000 boot file is the operating system initialization file that the computer launches upon the completion of the bootstrapping process?
   a. Ntldr
   b. Boot.ini
   c. Ntdetect.com
   d. Ntbootdd.sys
10. Which Windows 2000 boot file appears only on multiboot systems?
    a. Boot.ini
    b. Ntdetect.com
    c. Ntbootdd.sys
    d. Bootsect.dos
11. Osloader.exe replaces the functions of Ntldr and Ntdetect.com and can be used on either Intel or Alpha systems. True or False?
12. Before the Ntoskrnl file is given control of the computer, several DLLs, the HAL, and other device drivers are loaded into memory. True or False?
13. What is the last component launched before the logon dialog box is displayed that prompts you to press Ctrl+Alt+Delete?
    a. Winlogon.exe
    b. Lsass.exe
    c. Ntoskrnl.exe
    d. Hal.dll

14. The Windows 2000 environment is divided into two sections: the user mode and the kernel mode. True or False?
15. Why does Windows 2000 have two operational modes? (Choose all that apply.)
    a. The two modes are used to allocate execution time based on importance and priority.
    b. The two modes prevent nonsystem code from interacting with hardware.
    c. The two modes allow core environment-maintaining operations to function as background tasks.
    d. The two modes maintain the integrity of the security system.
16. All processes in the user mode function within a virtual machine. True or False?
17. Which of the following are true of user-mode processes? (Choose all that apply.)
    a. They can address 4 GB of memory.
    b. They have direct hardware access.
    c. Executive services provide proxied resources.
    d. They can be of five application types.
18. Which application types supported by Windows 2000 are actually emulations based on another application subsystem? (Choose all that apply.)
    a. Win32
    b. Win16
    c. DOS
    d. POSIX
    e. OS/2
19. Which executive service is involved whenever a user attempts to access a resource, no matter what the resource is?
    a. IPC Manager
    b. Security Reference Monitor
    c. Virtual Memory Manager
    d. File System Manager
20. Although the Registry is the primary means of storing configuration data for Windows 2000, several text-based initialization files are required as well. True or False?
21. When it is necessary to make system changes, which of the following tools does Microsoft recommend? (Choose all that apply.)
    a. Active Directory Users and Computers
    b. Computer Management
    c. REGEDIT
    d. System applet

22. The Registry is a list of exceptions instead of an exhaustive collection of control parameters. True or False?
23. Which of the following are true in regard to the Registry? (Choose all that apply.)
    a. A single change can cause system failure.
    b. A backup of the Registry is the only protection against invalid values.
    c. The Registry is self-repairing.
    d. Changes to the Registry take place only after a restart of the system.
24. The base functionality of network communications includes the ability to interact with other systems over physical connections as well as temporary remote access connections. True or False?
25. The base functionality of memory allocation involves which of the following items or components? (Choose all that apply.)
    a. Virtual Memory Manager
    b. Real-mode device drivers
    c. Virtual machines
    d. Process priorities

HANDS-ON PROJECTS

Project 3-1

To explore the Registry via REGEDT32:

1. Open the Run dialog box (**Start, Run**).
2. Type **REGEDT32**. Click **OK**.
3. Select **Read Only Mode** from the **Options** menu.
4. Select the **HKEY_LOCAL_MACHINE** subtree window.
5. Double-click the **Software** subkey.
6. Locate and double-click the **Microsoft** subkey. See Figure 3-10.
7. Locate and double-click the **Windows NT** subkey.
8. Double-click the **CurrentVersion** subkey.
9. Locate and select the **WinLogon** subkey.
10. Locate and double-click the **DefaultUserName** value entry.
11. Because you are in read-only mode, a message appears stating that your changes will not be saved. Click **OK**.
12. You see the data of the DefaultUserName value entry. Click **OK**.
13. Close REGEDT32 by selecting **Exit** from the Registry menu.

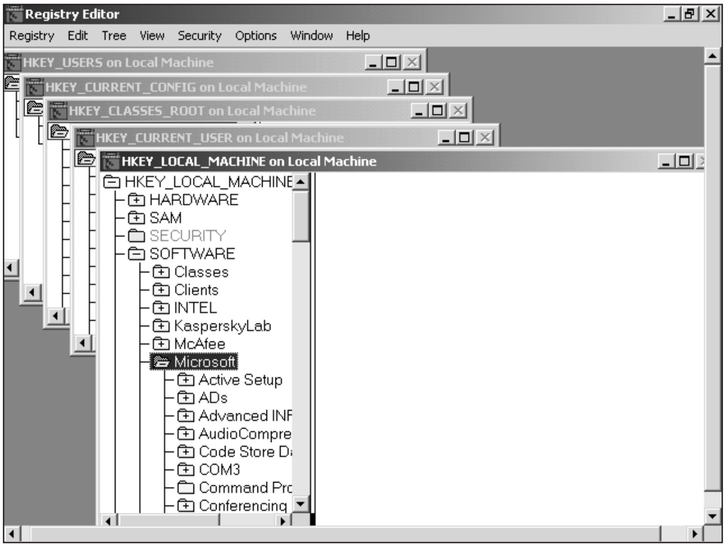

Figure 3-10 Exploring the HKEY_LOCAL_MACHINE\Software\Microsoft subkey

Project 3-2

To explore the Active Directory Users and Computers tool:

1. Open the Active Directory Users and Computers tool (**Start**, **Programs**, **Administrative Tools**, **Active Directory Users and Computers**).
2. Double-click the node named after your domain. (It should have a .local extension.)
3. Click the **Users** node. See Figure 3-11.
4. Double-click the **Administrator** user account in the Details pane.
5. Explore the multitabbed user properties dialog box, but do not make any changes.
6. Click **Cancel** when complete (this action discards your changes if you made any).
7. Select the **Computers** node. All nondomain controller systems are displayed in the Details pane.
8. Select the **Domain Controllers** node. All domain controller systems are displayed in the Details pane.
9. Click the **Builtin** node. All of the existing groups are displayed in the Details pane.
10. Double-click the **Administrators** group. Explore the multitabbed group properties dialog box, but do not make any changes.
11. Click **Cancel**.
12. Close the Active Directory Users and Computers tool by selecting **Exit** from the **Console** menu.

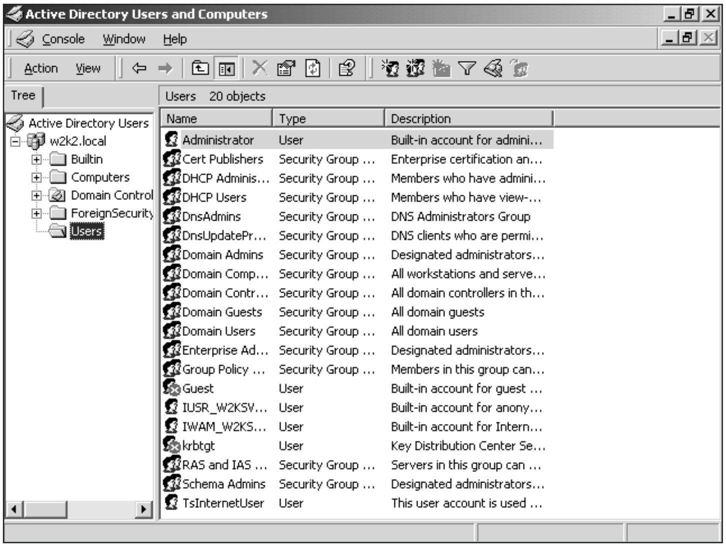

Figure 3-11 Exploring the Active Directory Users and Computers tool

Project 3-3

To explore the Computer Management tool:

1. Open the Computer Management tool (**Start**, **Programs**, **Administrative Tools**, **Computer Management**).
2. Click on the plus signs to the left of the **System Tools**, **Storage**, and **Services and Applications** nodes to expand their contents (if not already expanded). See Figure 3-12.

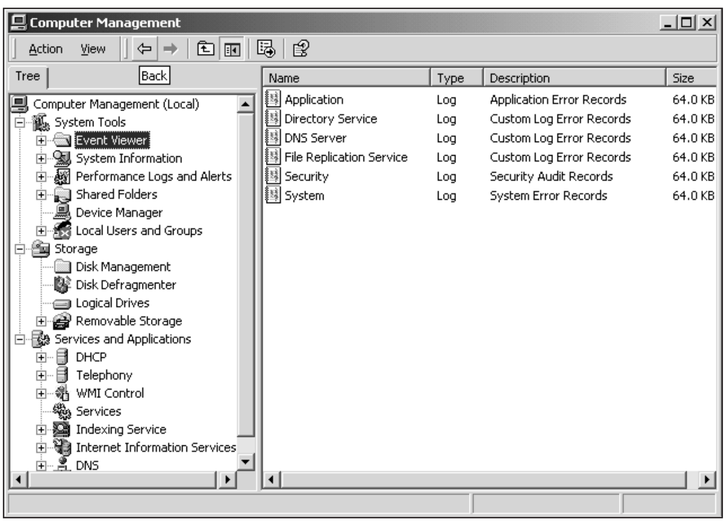

Figure 3-12 Exploring the Computer Management tool

3. Select the **Event Viewer** below the **System Tools** node.
4. Double-click the **System Log** in the Details pane.
5. Double-click an event detail.
6. After reading the detail, click **OK** to close it.
7. Select the **System Information** node below the **System Tools** node.
8. Double-click **Hardware Resources**, and then **IRQs**. This choice shows the IRQs currently in use. See Figure 3-13.

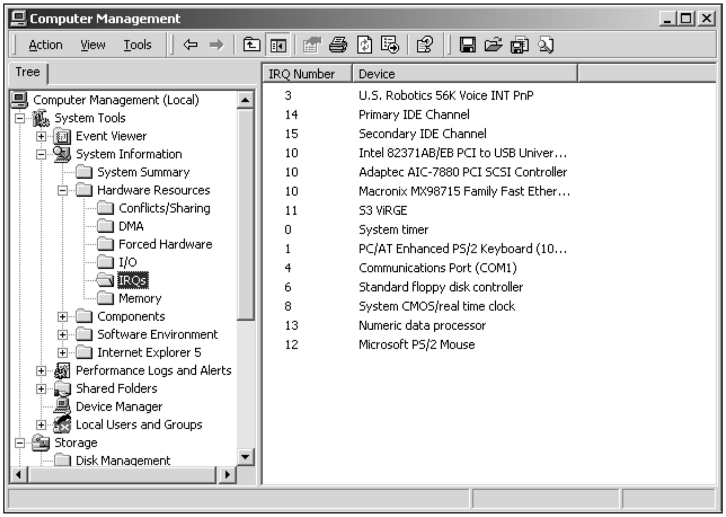

Figure 3-13 Using the Computer Management tool to determine the IRQs in use

9. Select the **Shared Folders** node.
10. Double-click the **Shares** detail. It reveals all of the current shares from this system.
11. Select the **Device Manager** node. It shows the categorized list of installed devices.
12. Select **Disk Management** below the **Storage** node. It displays the physical configuration of hard drives.
13. Select the **Disk Defragmenter** node. It is used to initiate defragmentation.
14. Select the **Logical Drives** node. It lists all local drives and mapped shares.
15. Select **Services** below the **Services and Applications** node. It lists all of the installed services and their current status.
16. Close **Computer Management** by clicking the **Close** button in the top-right corner of the window.

Project 3-4

To explore the Task Manager:

1. Launch the **Task Manager** (right-click over an empty area of the taskbar, then select **Task Manager** from the resulting menu).
2. Select the **Applications** tab. It lists all active applications and their status. See Figure 3-14.

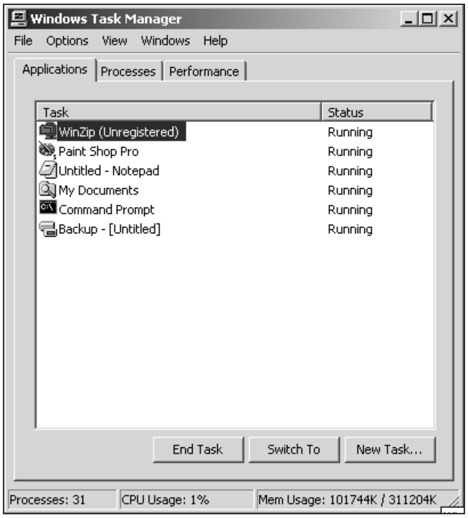

Figure 3-14 The Windows Task Manager

3. Select the **Processes** tab. It lists all active processes and related data such as CPU time, process ID, and memory usage. This tab can display more than a dozen different data items for each process.
4. Notice some of the core system files listed, such as Winlogon and Lsass.
5. Select the **Performance** tab. It displays real-time data about CPU and memory usage.
6. Close the Task Manager by selecting **Exit Task Manager** from the **File** menu.

Project 3-5

To explore the Control Panel:

1. Open the Control Panel (**Start**, **Settings**, **Control Panel**).
2. Double-click the **Add/Remove Hardware** applet to open it.
3. Click **Next** on the Welcome screen.
4. Select **Add/Troubleshoot a device**. Click **Next**.
5. After the wizard completes its search for devices, click **Cancel** to exit the wizard.
6. Double-click the **Add/Remove Programs** applet to open it. Notice the list of installed applications. See Figure 3-15.

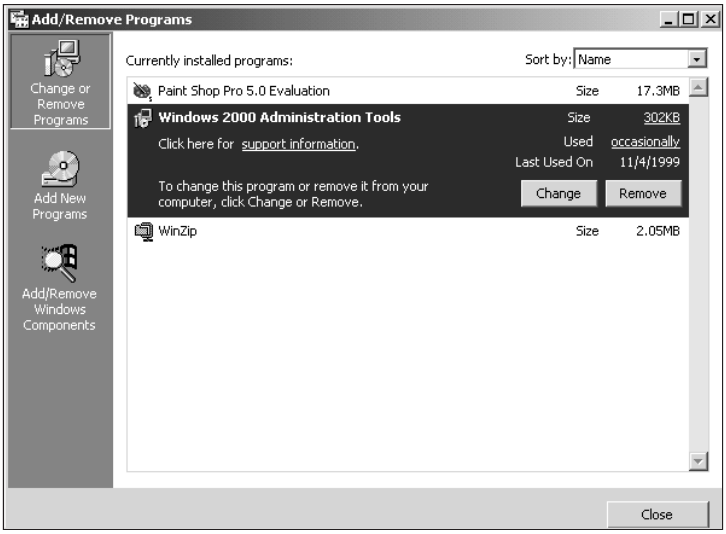

Figure 3-15 The Add/Remove Programs applet

7. Select the **Add New Programs** item. Notice the controls used to launch installation of a new application.
8. Select the **Add/Remove Windows Components** item.
9. Scroll down the list of installed and available Windows components.
10. Click **Cancel** to exit this wizard.
11. Click **Close**.
12. Double-click the **System** applet to open it.
13. Select the **Hardware** tab. Notice the links to install new hardware, access the Device Manager, and configure hardware profiles.
14. Select the **Advanced** tab. Notice the links to performance options, environmental variables, and start-up and recovery options.
15. Click **Cancel**.
16. Close the Control Panel by selecting **Close** from the **File** menu.

Hands-On Project

Project 3-6

To back up the Registry:

1. Launch the Registry Editor (**Start**, **Run**, **REGEDIT**).
2. Select **Export Registry File** from the Registry menu.
3. Provide a location to which to export the file.
4. Provide a name for the exported file. "W2Kserver-Registry" was chosen in Figure 3-16.

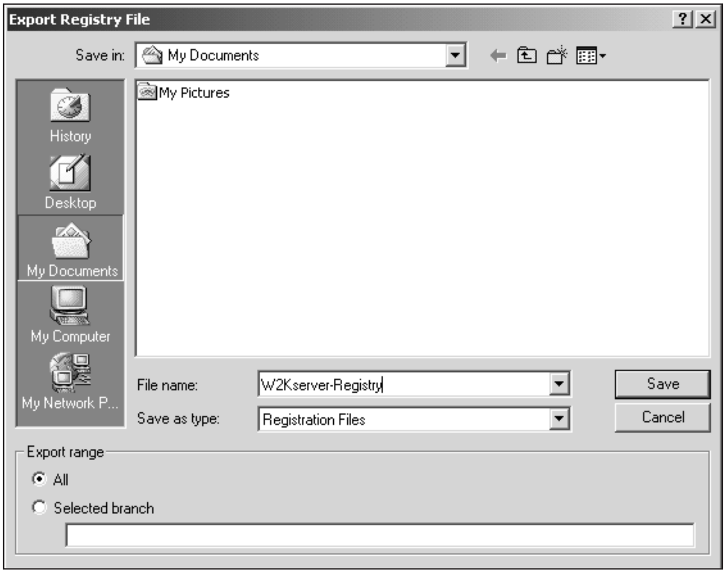

Figure 3-16 Exporting the Registry using REGEDIT

5. Select the **All** radio button in the **Export range** field. Click **Cancel**.
6. Select **Exit** from the Registry menu to close the Registry Editor.

CASE PROJECTS

1. Describe the files involved with the initialization of the Windows 2000 operating system that are active between bootstrapping and the prompt to log on.
2. Describe at least five basic functions of an operating system and explain how Windows 2000 demonstrates these functions.
3. Describe the process of bootstrapping and the problems that can occur before an operating system is launched.
4. Describe the Windows NT Registry. Be sure to mention its structure, purpose, layout, fault-tolerant mechanisms, and editing practices.